Proceedings of the 2004 IEEE
Conference on Robotics, Automation and Mechatronics
Singapore, 1-3 December, 2004

A Divide and Conquer Algorithm for Rectilinear Region Coverage

Amit Agarwal, Meng-Hiot Lim, Lip Chien Woon
Intelligent Systems Center, Techno Plaza
Nanyang Technological University
Singapore 639798
{pg0212208t, emhlim, 810301075597}@ntu.edu.sg

***Abstract*—We give an algorithm to generate a coverage motion plan for a single unmanned reconnaissance aerial vehicle (URAV) over a holed rectilinear polygonal region. The URAV is equipped with a stabilized downward-looking sensor which has a square footprint of a fixed area. The algorithm is based on the principle of divide-and-conquer. In the first step, we generate the lexicographically maximum area rectangle partition of the polygon. We outline a sweep line based algorithm to compute this partition. Next, we find a coverage motion plan for movement of the sensor over each rectangle in the partition. Plans for adjacent rectangles are finally merged to generate a complete plan for the entire region. Our algorithm yields paths with minimal length and minimal turns.**

Keywords—*motion planning, region coverage; polygon partitioning*

I. INTRODUCTION

One of the fundamental problems in the use of robots to autonomously perform tasks in physically or geographically distributed environments is planning the motion (generally, ideally, a tour) of the robot to the sites of interest. One standard formulation of this problem requires finding a set of minimum length paths, each of which represents an alternative tour that enables the robot to visit each site and get back to its default location. Minimizing the number of turns may be included as an additional objective. This is particularly useful in problems such as planning the motion of a factory robot. Unit turn costs in such cases are often higher than the unit linear travel cost. The minimum length tour problem and its variations have been extensively studied and are generally known as the traveling salesman problem (TSP) [1, 2, 3, 4, 5].

The traditional TSP formulation is however not an appropriate model for problems involving coverage of a region or a set of regions in which the inter-region traversal cost is of the order of the cost of scanning the regions themselves. This is mainly due to an inability to treat the underlying topology as a set of points, each representing a region in/over which an optimum motion plan can be trivially obtained, connected by weighted segments. Problems such as spray painting car body plates, automated vacuum cleaning of factory floors, high speed surface milling and reconnaissance using low-flying aerial vehicles fall in the category of region covering problems.

Covering problems involving minimization of the number of turns executed and the travel distance on even planar rectilinear regions are NP complete in the general case (for example, see [6]). Due to this, scalable efficient algorithms that treat the region to be covered as an undivided whole will likely remain elusive (unless P=NP). Thus, an alternative solution strategy is needed. In this work, we describe a divide-and-conquer algorithm for covering a rectilinear polygon. More specifically, our algorithm is invariant to the presence of hole(s) in the polygon. The underlying polygon need not be horizontally (or, vertically) convex. Henceforth, except in the Section II, unless stated otherwise, the terms 'holed rectilinear polygon' and 'polygon' are used interchangeably and both refer to a simple holed rectilinear polygon.

Our region coverage problem arises in the context of airfield or urban zone surveillance. The region of operation is assumed known *a priori*. We assume that the altitude of operation of the URAV is fixed. The flight altitude must guarantee, amongst other matters, the desired minimum resolution of images given the focal length limitation of the onboard image acquisition system. The footprint of the sensor of the URAV on the ground (assumed flat) is square-shaped and has a fixed area A_s. It is also assumed that during the scan, the scanning device moves parallel to the plane in which the region to be scanned is embedded.

We give an algorithm to compute a path along which the geometric centroid of the sensor footprint can be moved to enable a complete scan of the rectilinear polygon with minimal repetitions and number of turns. We assume that the motion plan for the URAV can be obtained using the motion plan for the sensor footprint. If the URAV can perform pivotal turns of 90 or 270 degrees then the path generated by our algorithm can restrict the footprint to within the closed polygon. Else, the footprint will violate the boundary of the polygon while executing certain turns. This violation can be traded off against the requirement for complete scanning. This engineering tradeoff is dependent on the particular application (For example, in autonomous lawn mowing, in order that the lawn mower blades do not damage the cultivated plants on the periphery, it is imperative that the blades do not violate the lawn boundaries. Mathematically complete coverage in the case of autonomous lawn mowing is therefore undesirable.).

This section is immediately followed by a section on some pertinent definitions. Section III presents our motivation in studying the minimum length, minimum turn problem for coverage. Section IV outlines the stages of our algorithm. Finding a lexicographically axis-parallel rectangle partition is a sub problem of our algorithm for the sensor footprint motion

0-7803-8645-0/04/$20.00 © 2004 IEEE

planning problem. A sweep line technique for this problem is outlined in Section V. We conclude with Section VI.

II. DEFINITIONS

This section provides definition of geometric terms used here. Some of these may not be consistent or, equally inclusive or exclusive as definitions of some of these terms found elsewhere.

- Polygon – A polygon $\wp$ is a connected sequence of line segments that partitions the plane into one or more finite, closed space(s) and exactly one open space.
- Simple polygon – A simple polygon is a polygon with segments whose pairwise intersection is either a null set or exactly one point that lies on one of the end points of each segment belonging to the intersecting pair. In addition, the intersection of any two pairwise intersecting segments is an empty set.
- Interior of a polygon – The intersection of all possible sets of which the polygon is a subset is called the interior of the polygon.
- Polygon with hole – A polygon with holes is the intersection of a polygon with the negation of a polygon in its interior. The universal set being the infinite plane in which the polygons are embedded.
- Rectilinear (orthogonal) polygon – A polygon whose nonparallel segments are at 90° or 270°.
- Decomposition of a polygon – It is a set of polygons whose union is the polygon itself.
- Rectangle cover of a polygon – It is a set of rectangles comprising at least two members whose intersection is nonzero and whose union is the polygon itself. A rectangle cover may include rectangles defined with the aid of Steiner points. A polygon may or may not have a rectangle cover.
- Rectangle partition of a polygon – It is a set of rectangles whose pairwise intersection is either a null set or a subset of the union of exactly one segment of each of the intersecting rectangles and whose union is the polygon itself. Not all polygons entail a rectangle cover.
- Lexicographically maximum area rectangle partition of a polygon ($\Re_{pM}$) – Let $\Re_{pM}$, be a rectangle partition of $\wp$. Let $|\Re_{pM}| = n$, $\Re_{pMi} \in \Re_{pM}$ and areas of rectangles be $A(\Re_{pM_1}) \geq A(\Re_{pM_2}) \geq A(\Re_{pM_3}) \geq \ldots \geq A(\Re_{pM_n})$. Let $\{\Re_p'\}$ be the set of all other rectangle partitions of $\wp$. $\Re_{pM}$ is a lexicographically maximum area rectangle partition of $\wp$ iff $\forall \Re_k \in \{\Re_p'\}$ either $A(\Re_{pM_1}) > A(\Re_{k_1})$ or $A(\Re_{pM_{i+1}}) \geq A(\Re_{k_{i+1}})$ (if it exists) for i = 1 through $(n-1)$ for partitions for which $A(\Re_{pM_i}) \not> A(\Re_{k_i})$.

More information on lexicographically maximum rectangle partition of polygons is presented in Section V.

- Cycle in a rectangular grid – A cycle C in a rectangular grid is a closed path through centroids of all grid cells.
- Adjacent cycles – Two cycles are adjacent iff they *a.* are space filling curves in adjacent rectangles. *b.* have parallel segments of minimum length equal to twice the width of the sensor footprint within a perpendicular distance of one sensor footprint.

The readers may note that it is trivial to obtain a cyclic path by merging two adjacent cycles. Cycle union may lead to overlapping coverage of strip of thickness less than the width of the sensor footprint.

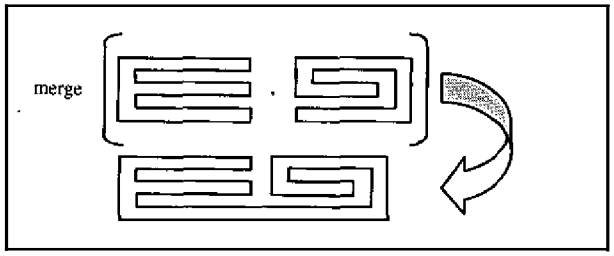

Figure 1. Cyle Merge Operation.

Note: in Figure 1, both C_1 and C_2 have optimum number of turns. The merge operation results in a decrease of 4 turns. Also, the distance between consecutive parallel lines is equal to the width of the sensor.

III. MOTIVATION FOR POLYGON PARTITIONING

We are interested in determining an efficient motion plan for a sensor's footprint to completely cover a region that can be appropriately modeled as a planar simple rectilinear polygon. The motion plan for the sensor will largely determine the motion of the URAV on which the sensor is mounted. We assume that the polygonal region has been completely identified prior to the beginning of the scanning process. By complete identification we mean we have an edge list or a doubly connected edge list representation of the polygon graph. Either representation may be obtained from the other in $O(N)$ steps [7]; N is the number of vertices in the polygon. In our work, the ratio of sensor footprint area A_s to the areas of uninteresting projections in the region of interest or to the area of the hole(s) is very small. If this is not true then, one can trivially plan the motion using a lapping motion primitive on a bounding rectangle of $\wp$.

A natural question of considerable interest in region cover problems is the amount of time needed to complete the task. The length of path traversed and the total number of turns made are critical determinants of this time. Turns, in addition, impose an additional cost in terms of introducing non-linear errors in INS measurements. Thus, we desire a motion plan that has minimum length and involves minimum number of turns. However, the problem is NP-hard and we instead seek optimal solutions using a divide-and-conquer approach.

Our approach is to decompose the rectilinear polygon into its lexicographically maximum area rectangle partition. Next, we overlay a grid on each rectangle in the partition (see Section IV). Following this, we attempt to find space-filling closed-path curves (Hamiltonian cycles) in each gridded rectangle. Cycle merging operation is applied to each pair of adjacent cycles (whenever the pair exists) to obtain a complete motion plan.

Partitions with small cardinality are desirable for two reasons – *a*. A total of $C-1$ ($C=|\mathcal{C}|$) cycle merge operations must be performed to obtain the complete coverage plan. *b*. The number of turns in the complete coverage plan can be potentially reduced by reducing the number of cycles to be merged (see Theorem 3.1). While several efficient algorithms for minimum cardinality partitions are known [9, 10], we constructed the partition by recursively extracting the largest rectangle from $\wp$. One of the reasons for doing this is that its temporal cost is $O(N^2 \log N)$ (see Section V) as opposed to $O(N^2\sqrt{N})$ for minimal cardinality partitions. Also, by Theorem 3.2, for random rectilinear polygons, the cardinality of lexicographically maximum area rectangle partition is within a constant factor of the cardinality of the minimum rectangle partition.

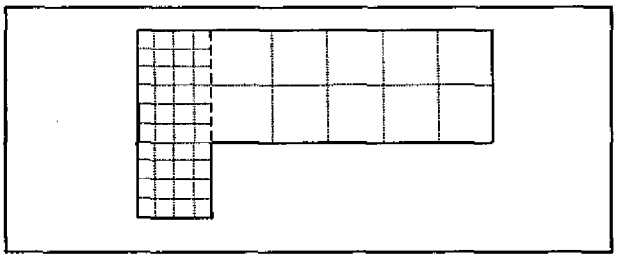

Figure 2. A grid partition of $\wp$.

Theorem 3.1: A cycle cover of a unit grid graph with C cycles can be converted into a Hamiltonian cycle in the graph that has a maximum of $2C-2$ additional turns.

Proof: See Theorem 4.2 [6].

Theorem 3.2: The upper bound on the cardinality of lexicographically maximum area rectangle partition is within a constant factor of the upper bound on the cardinality of minimum rectangle partition.

Proof Idea: A minimum rectangle partition has $O(N)$ rectangles. Each rectangle in the lexicographically maximum area rectangle partition is defined by up to four Steiner points. In case a rectangle is defined by four Steiner points, it must include at least four vertices of the rectilinear polygon on its edges. Thus, the net change in the number of vertices in $\wp$ due to the extraction of a rectangle defined by four Steiner points is greater than or equal to zero. Any rectilinear polygon can have only $O(N)$ such rectangles. If three or less Steiner points are used to define a maximum area rectangle, the total number of vertices in $\wp$ upon the extraction of the rectangle decreases by at least one. Each vertex (Steiner or otherwise), can be shared by exactly one rectangle if the interior angle at the vertex is 90°. If the interior angle is 270°, it can be shared by a maximum of three rectangles. With the words 'interior' and 'exterior' mutually exchanged, this property also holds true for vertices belonging to the polygon(s) that define the holes. Thus, the statement of Theorem 3.2 follows.

Lemma 3.1: The cardinality, $|\Re_M(\wp)|$, of minimum partition of a rectilinear polygon into axis parallel rectangles, is either less than or equal to the cardinality, $|\Re_L(\wp)|$, of the lexicographically maximum area rectilinear rectangle partition.

Proof: It follows from the definition of minimum cardinality rectangle partition that $|\Re_M(\wp)| \not> |\Re_L(\wp)|$. For the remaining part of the proof, see figure 4.

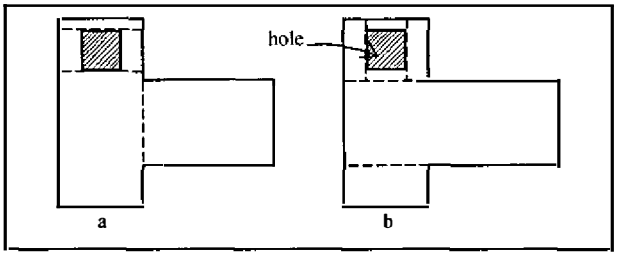

Figure 3. (a) A minimum cardinalty partition ($|\Re_M(\wp)|=5$). (b) The lexicographically maximum partition ($|\Re_L(\wp)|=6$).

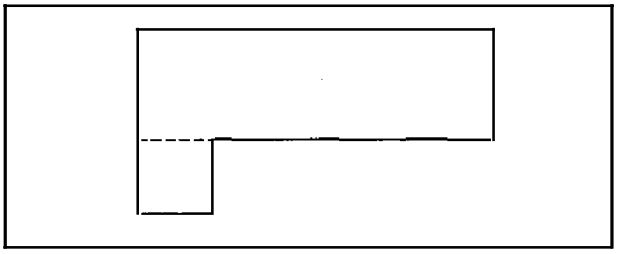

Figure 4. A rectilinear polygon $\wp \ni |\Re_L(\wp)| = |\Re_M(\wp)|$.

IV. COVERAGE ALGORITHM FOR RECTILINEAR POLYGONAL WORKSPACE

The algorithm for sensor footprint motion planning is as follows.

STEP1: Find the lexicographically maximum area rectangle partition of the rectilinear polygon.

STEP2: Overlay on each rectangle the smallest rectangle grid whose cells have the same dimension as that of the sensor footprint.

STEP3: For rectangular grids with odd cardinality, do a minimum Hamiltonian node completion.

STEP4: Find a cycle cover of each rectangle.

STEP5: Merge cycles in adjacent rectangles.

END

A space filling Hamiltonian curve in $\wp$ generated by the motion of nonoverlapping sensor footprint (if it exists), by definition, minimizes the distance traveled while ensuring a complete scan. However, the problem of finding such a tour with a minimum number of turns is NP complete. On the contrary, both problems can be trivially solved for rectangular regions. Hence, a rectangular partition is generated in STEP1. The edges of rectangles constituting the partition are either orthogonal or collinear with the edges of $\wp$.

A coverage motion plan for each rectangle in the partition must take the geometric centroid of the sensor footprint through the center of each cell of the grid overlaid on the respective rectangle. Only 4-adjacent motions are permitted. A minor problem with the approach in STEP2 is that grid lines in any rectangle, in a general case, will be out of alignment with grid lines in any other rectangle in the partition. This necessitates that non-orthogonal turns be made in the region of merger between adjacent cycles. A complete treatment of this problem will be discussed in a later work.

Hamiltonian cycle does not exist in planar gridded rectangles with odd number of cells. STEP3 ensures the presence of a Hamiltonian cycle by augmenting the rectangles minimally; for example, by adding either a row or a column of cells. This procedure however increases the path length.

Once the existence of a Hamiltonian space filling curve is guaranteed in STEP3, a Hamiltonian space filling curve with minimum number of turns can be trivially found for each rectangle. Following this, adjacent cycles are merged. Each merge can introduce up to two additional turns. By merging two cycles, it is possible to obtain a cycle with number of turns up to four less than the sum of the number of turns in the two cycles (see Figure 1). We however, do not address the problem of optimally merging the cycles in this work.

If a rectangular bounding box of the robot whose orthogonal axes of symmetry are aligned with the axes of symmetry of the scanner footprint has smaller length and breadth as compared to those of the scanner footprint then, a Hamiltonian space filling curve derived above will always constitute a valid path for the robot too. This has an important implication in applications such as spray painting or vacuum cleaning of factory floors where the robot is likely to be constrained by obstacles or no-go zones.

V. ALGORITHM FOR COMPUTING LEXICOGRAPHICALLY MAXIMUM AREA RECTANGLE PARTITION

Our method of computing the lexicographically maximum area axis-parallel rectangle partition of a holed rectilinear polygon is based on recursive extraction of the largest empty axis-parallel rectangle from the polygon and its intersection with the negation of the hitherto extracted rectangle (if any). One way of doing this is a simple extension of the algorithm in [11]. Combining Theorem 3.2 with the results in [11] gives a time-bound of $O(N^2 \log^2 N)$ for the extended algorithm.

In the remaining part of this section we outline a new approach to compute the partition. Our approach is based on a sweep line technique which involves moving a (d–1)-dimension hyperplane parallel to the d-axis [8].

The input polygon may be in any orientation. However, before applying the sweep line algorithm, the orthogonal polygon must be rotated to make its edges axis-parallel. Without loss of generality, the sweep line moves discretely from top to bottom, stopping only at the polygon vertices, called sites. The movement of the sweep line to a new site is called a site event. At each site event, new rectangle record(s) is/are made or the complete definition of previously registered rectangle record(s) is/are achieved and the area(s) is/are computed. The area of the largest rectangle found yet is maintained in the memory. A sweep comprises $N_V(\wp)$ site events; $N_V(\wp)$ is the vertical complexity of the polygon $\wp$. When the sweep line reaches the last site in the site event queue, the algorithm terminates and the largest empty axis-parallel rectangle is obtained.

Due to limited space, data structures needed for efficient implementation of the algorithm and statements concerning the space and time bounds and their proofs are presented elsewhere [12].

The algorithm is as follows –

SWEEPLINE ALGORITHM FOR FINDING LARGEST EMPTY AXIS-PARALLEL RECTANGLE IN ORTHOGONAL POLYGON

1: Sort the vertices of the polygon along y-axis.

2: Move the sweep line from top to bottom.

3: At each site event,

3a: Create rectangle records by storing the horizontal edge(s) that is/are incident on the site(s) in the site event.

3b: For each horizontal edge corresponding to the site event, check if it overlaps with the y-projection of its immediate parent(s). If yes then, terminate the rectangle(s) generated by those parent(s). Compute their area(s).

3c: Check for new rectangles that may begin from above the sweep line.

4: Output the largest empty rectangle.

5: $\wp = \wp - (\text{largest empty rectangle})_i$

6: While $N_V(\wp) > 4$, repeat steps one through 5.

7: Output lexicographically maximum area rectangle partition of $\wp$.

END

Figure 5 shows a typical polygon whose lexicographically maximum area rectangle partition we wish to compute. First, the largest empty axis-parallel rectangle is computed. Figure 6 shows the result of application of stages one through five of our algorithm. The complete execution of our algorithm yields

the desired partition for the polygon in Figure 5 and is shown in Figure 7.

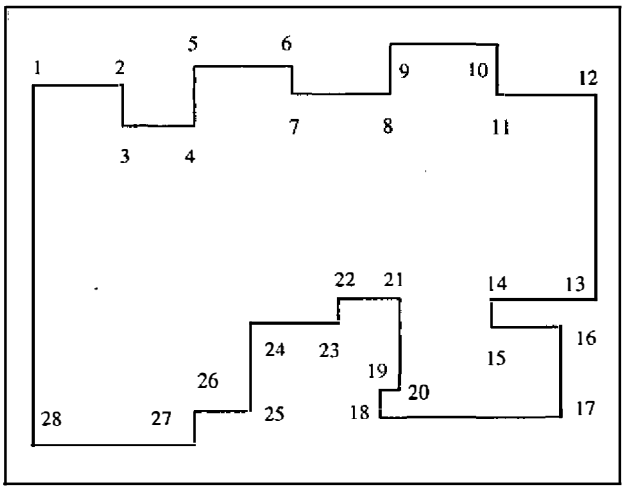

Figure 5. A rectilinear polygon.

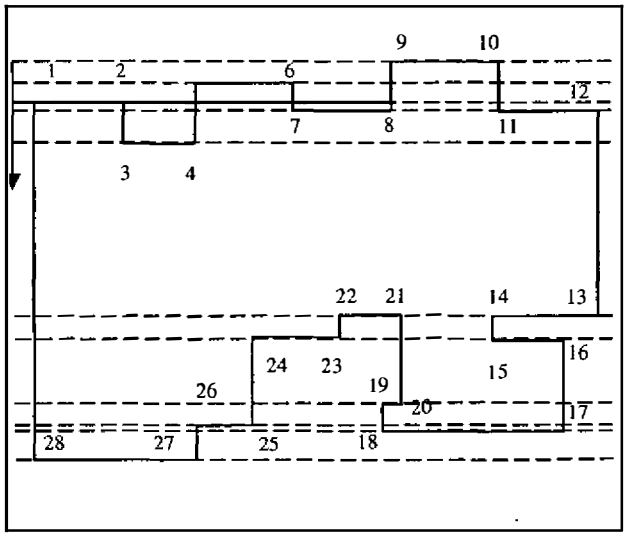

Figure 6. Snapshots of the sweep line at site events. A new rectangle record is created or an earlier rectangle record is terminated and the corresponding rectangle is completely determined only by site events.

Lemma 5.1: In infinite precision computing, the set of points defined by the largest axis-parallel rectangle in an orthogonal polygon is unique.

Proof: The proof follows from the observation that area of a rectangle (the dot product of adjacent sides a and b) is a monotonically increasing function of both a and b. In an infinite precision representation scheme, a and b are unique.

The sweep line algorithm together with Lemma 5.1 implies that all orthogonal polygons entail a lexicographically maximum rectangle that can be found in polynomial time.

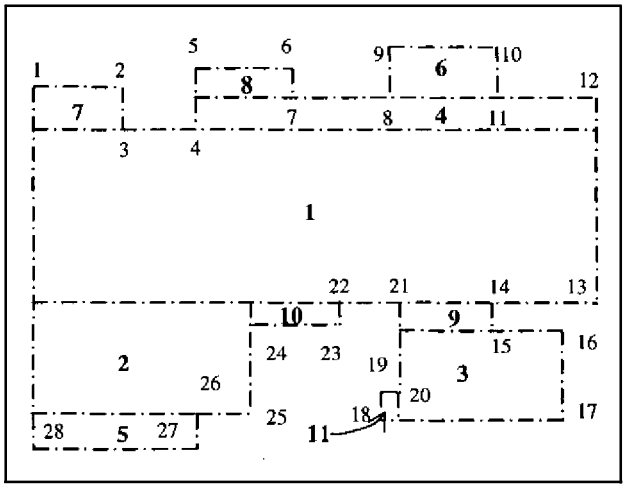

Figure 7. Lexicographically maximum rectangle partition of the polygon in Figure 5.

Lemma 5.2: In infinite precision computing, lexicographically maximum axis-parallel rectangle partition of an orthogonal polygon is unique.

Proof: The set of lexicographically maximum area rectangle can be obtained via a recursive application of the sweep line algorithm to $\{\{\wp\} - \bigcup_{k=1 \text{ to } i} \Re_{\max_k}\}$ where $\Re_{\max_i}$ is the maximum rectangle in the i^{th} recursion. Then lemma 5.2 follows from the proof of lemma 5.1.

Lemma 5.3: The maximum number of new rectangles whose top edge is above the sweep line at the site event that generate them is 2.

Proof Idea: A horizontal segment on the sweep line can create new rectangle(s) that begin from above the sweep line only if there exists an infinitesimally small vector that is parallel to the sweep line motion vector, incident on it and, lies within the convex space of $\wp$. In the decision tree rooted at this assertion, we have two mutually exclusive and exhaustive cases –

A. The immediate parent of the horizontal segment on the sweep line is a single segment. In this case, either no new rectangle above the sweep line can be generated (as in Figure 8a) or up to two new rectangles that begin from above the sweep line can be generated (as in Figure 8d).

B - There are two or more immediate parents, the union of whose x-intervals is a superset of the interval of the segment on the sweep line under current consideration (Figures 8b and 8c). In this case, either one or two new rectangles that begin from above the sweep line are generated.

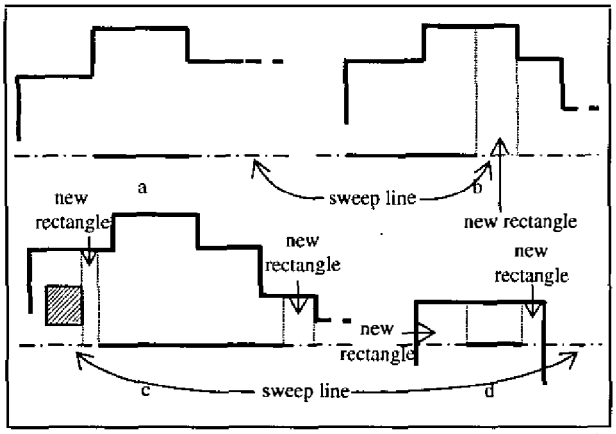

Figure 8. New rectangle records generated by the sweepline that correspond to rectangles that lie on either side of the sweep line that generates them.

Corollary 5.3.1: For a polygon $\wp$, $O(N_V(\wp))$ new rectangles which begin from above the sweep line that generates them are created by the sweep process.

This follows from lemma 5.3 and the observation that a polygon $\wp$ has $N_V(\wp)$ site events. The lower bound is zero as shown by the example in Figure 9.

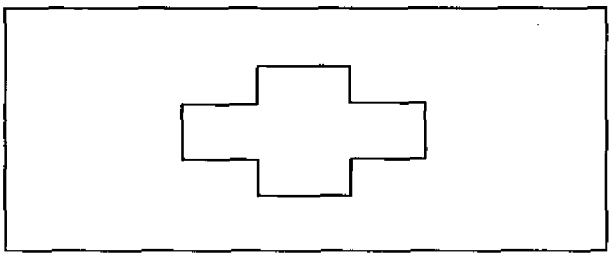

Figure 9. A polygon with non-zero vertical complexity in which no new rectangle above the sweep line is generated by a segment on the sweep line.

VI. CONCLUDING REMARKS

We presented a polynomial-time divide and conquer algorithm for the coverage of a rectilinear polygonal region by a downward looking stabilized sensor mounted on a URAV. The region can have uninteresting areas called holes. Our algorithm is suited for coverage when the area of the sensor footprint is small in comparison with the area of holes or uninteresting projections in the region of interest. Our algorithm yields a motion path for the sensor footprint that has minimal length and involves minimal number of turns. We introduced the concept of lexicographically maximum axis-parallel rectangle partition of a rectilinear polygon and outlined an algorithm for generating it.

REFERENCES

[1] T. Stützle and M. Dorigo. ACO algorithms for the traveling salesman problem, K. Miettinen, M. Makela, P. Neittaanmaki and J. Periaux (eds.), *Evolutionary Algorithms in Engineering and Computer Science*, Wiley, 1999.

[2] M. Grötschel and O. Holland "Solution of large scale symmetric traveling salesman problems", Mathematical Programming, 51, pp.141-202, 1991.

[3] M. L. Fredman, D. S. Johnson, L. A. McGeoch and G. Ostheimer, "Data Structures for Traveling Salesmen", J. of Algorithms, 18:3, pp.432-479, 1995.

[4] D. S. Johnson and L. A. McGeoch, "Experimental analysis of heuristics for the STSP", *The Traveling Salesman Problem and its Variations*, G. Gutin and A. Punnen (eds.), Kluwer Acad. Pub., Dordrecht, pp.369-443, 2002.

[5] D.Eppstein, "The traveling salesman problem for cubic graphs", 8th Workshop on Algorithms and Data Structures, Ottawa, LNCS 2748, pp.307-318, 2003.

[6] E.M. Arkin, M.A. Bender, E.D. Demaine, S.P. Fekete, J.S. B. Mitchell and S. Sethia, "Optimal covering tours with turn costs", Proc. of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms, Washington, D.C., United States, pp.138-147, 2001.

[7] F.P. Preparata and M.I. Shamos, *Computational Geometry an Introduction*, Springer Verlag New york Inc., 1985, pp.15-18.

[8] M. Van Kreveld, M. Overmars, O. Schwarzkopf, M. De Berg, (ed.), *Computational Geometry Algorithms and Applications*, Springer-Verlag, 2nd Revised Edn. 2000, pp.22-26.

[9] H. Imai and T. Asano, "Efficient algorithms for geometric graph search problems", SIAM J. on Computing, 15, pp.478-494, 1986.

[10] R. Chadha and D. Allison, "Partitioning rectilinear figures into rectangles", Proc. of the 16th Annual ACM Conf. on Computer Sci., Atlanta, USA, pp. 102-106, 1988.

[11] A. Aggarwal and S. Suri, "Fast algorithms for computing the largest empty rectangle", 3rd Annual ACM Symposium on Computational Geometry, Waterloo, Canada, pp. 278-290, 1987.

[12] A. Agarwal, M.H. Lim and M.J. Er, "On the construction of lexicographically maximum area partitions of general rectilinear polygons", *unpublished*.